REMARKS/ARGUMENTS

In the Office Action, the Examiner has rejected claims 7, 13, 19 and 22 under 35 U.S.C. § 103(a) as being unpatentable over *Peter Haggar* in view of US Patent No. 6,654,778 (*Blandy et al.*). Claims have also been rejected under 35 U.S.C. 112. These rejections are fully traversed below.

Rejection of claims under 35 U.S.C. 112

Fig. 3 of the present application depicts a method 300 for representing a Java object as a string in accordance with one embodiment of the invention. With reference to Fig. 3, the specification states that at operation 302, a reference to a Java object is pushed on the execution stack. Next, at operation 304, an inventive Java Bytecode operation is executed. The inventive Java Bytecode operation is designated to represent the object as a string. It should be noted that the Java object is referenced by the reference pushed on the execution stack (operation 302). The specification further states that at operation 306, the string representation of the Java object is determined using the reference to the Java object.

Referring to Fig. 2A, a reference to a Java object (reference A) is depicted in execution stack 212. The specification states that when the inventive Java "to-string" Bytecode instruction 208 is executed, the string representation of the Java object referenced by reference A is determined (Specification, paragraph 24).

It is respectfully submitted that those skilled in the art know that a reference to an object can be used to access that object. Further, one or more fields of the object can be accessed by using the reference (see, for example, fields 1, 2 and 3 of a Java object 210 shown in Fig. 2A of the present application). As such, when a field is accessed, binary data of the field (e.g., "100") can be accessed and interpreted in accordance with various conventions. For example, binary value "100" can represent an integer or a character (e.g., "4", or "A"). As also noted in the

specification and known in the art, often there is a need to represent a Java object as a string of characters. For example, in order to print an integer object, (or an integer field of an object) there is a need to represent the integer as a string of characters (Specification, paragraph 9).

The specification states and those skilled in the art well know that Java programs are generally platform independent (Specification, page 2). As such, an actual string representation may be dependent on the particular virtual machine implementation and/or the hardware and/or operating system. Thus, one of skilled in the art will know that when an object is accessed, the digital data stored for a field of the object can be represented in accordance with a particular standard (e.g., ANSI, EBCIDIC) that may apply to the particular hardware and/or operating system that runs the Java virtual machine (Examples of common character representations are provided herewith in an Information Disclosure Statement for the Examiner's convenience). Accordingly, it is respectfully submitted that one of ordinarily skilled in the art knows that an object and/or a field of an object can be represented as a string of characters (e.g., "10," "Hello World") in accordance with various known standards (e.g., ANSI, EBCDIC). However, conventionally, a Java method is invoked by the virtual machine to represent objects as string characters (Specification, paragraph 9). The claimed invention pertains to a Java Bytecode instruction that effectively determines a string representation of a Java object by using a reference to the Java object stored on an execution stack.

Rejection of the claims under 35 U.S.C. 102

In the Office Action, the Examiner has asserted that the "getfield" Bytecode described by *Haggar* teaches a Bytecode instruction that determines a string representation associated with a Java object (Office Action, page 7). It is noted that *Haggar* states that a "getfield" opcode is used to fetch a field from an object. More particularly, when a "getfield # 5" is executed, "the top value from the stack is

popped," "then the "# 5" is used to build an index into the runtime constant pool of the class where the reference to name is stored. When this reference is fetched, it is pushed onto the operand stack" (*Haggar*, page 3).

However, it is respectfully submitted that the "getfield" operation described by *Haggar* merely fetches a field from an object. As such, the "getfield" opcode does NOT additionally determine a string representation for the fetched field. Accordingly, it is respectfully submitted that the Examiner's rejection of claim 1 under 35 U.S.C. 102(a) is improper and should be withdrawn.

Based on the foregoing, it is submitted that the claims are patentably distinct over the cited art of record. Additional limitations recited in the independent claims or the dependent claims are not further discussed because the limitations discussed above are sufficient to distinguish the claimed invention from the cited art. Accordingly, Applicant believes that all pending claims are allowable and respectfully requests a Notice of Allowance for this application from the Examiner.

Applicants hereby petition for an extension of time which may be required to maintain the pendency of this case, and any required fee for such extension or any further fee required in connection with the filing of this Amendment is to be charged to Deposit Account No. 500388 (Order No. SUN1P843). Should the Examiner believe that a telephone conference would expedite the prosecution of this application, the undersigned can be reached at the telephone number set out below.

Respectfully submitted,
BEYER WEAVER & THOMAS, LLP

/RMahboubian/
R. Mahboubian
Registration No. 44,890

June 15, 2006

P.O. Box 70250
Oakland, CA 94612-0250
(650) 961-8300